[illegible]

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/992,914	11/14/2001	Craig Russ	TN241	6966

7590 03/24/2005

Lise A. Rode
Unisys Corporation
Unisys Way, MS/E8-114
Blue Bell, PA 19424

EXAMINER	
CHU, GABRIEL L	

ART UNIT	PAPER NUMBER
2114	

DATE MAILED: 03/24/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

[illegible]

Office Action Summary	Application No.	Applicant(s)	
	09/992,914	RUSS ET AL.	
	Examiner	Art Unit	
	Gabriel L. Chu	2114	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *14 November 2001*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-29* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-29* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)	4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)	5) ☐ Notice of Informal Patent Application (PTO-152)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.	6) ☐ Other: _____.

DETAILED ACTION

Claim Objections

1. Claims 1 and 22 are objected to because of the following informalities: Referring to claims 1 and 22, the use of quotation marks obscures the intended meaning and may imply the word is not being used in its accepted sense. For the purpose of examination, the term " 'isolated' " is understood to refer to "isolated" without quotation marks. Appropriate correction is required.

Claim Rejections - 35 USC § 103

2. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

3. **Claims 1-29 are rejected under 35 U.S.C. 103(a) as being unpatentable over US 5781770 to Byers et al. in view of US 6243860 to Holland.** Referring to claim 1, Byers discloses a method for maintaining system integrity in a multiple user environment (From line 35 of column 2, "LAN".), the method comprising: marking a first procedure isolated (From the abstract (with emphasis), "A processor shelf controller monitors processor shelf status. On shutdown or reset request either initiated by a central controller or the processor shelf, the processor shelf controller provides at least a first timer that allows the operating processor shelf sufficient time to shutdown operating system **processes**." Wherein the process(es) is "made visible" with specificity from others, see lines 7-36 of column 6.), and in response to an external command

associated with the first procedure, allotting a predefined period of time for the first procedure to complete before executing the external command (From the abstract, "A processor shelf controller monitors processor shelf status. On shutdown or reset request either initiated by a central controller or the processor shelf, the processor shelf controller provides at least a first timer that allows the operating processor shelf sufficient time to shutdown operating system processes. For graceful shutdowns, a second timer is provided to allow the operating processor shelf to clear all application related processes prior to start of the first timer.").

Although Byers does not specifically disclose this marking a first procedure isolated is marking a first procedure associated with a first stack, wherein the first procedure is declared by a second procedure isolated (Wherein such marked as isolated may also be interpreted as allocation of memory, as shown in Holland as follows.) with a second stack, calling a child process by a parent is well known in the art. An example of this is shown by Holland, from line 9 of column 6 (with emphasis), "FIG. 3 shows the call flow for the compiler 12 of FIG. 1. First, at 40, the exemplary C external routine is called, for example, from the parent process 6 which provides a processor( ) shell port. Next, within the exemplary C context, at 42, the routine maintains a program stack of blocks to manage the transient execution of a plurality of child processes, such as 8,38 of FIG. 2B. The stack is employed because a parent process can call a child process, and that child process, in turn, can spawn other child processes using nested calls. As such, the program stack maintains **generated block information** so that program control can be returned in an orderly manner as child processes terminate, thereby returning to the respective parent processes. Then, after saving any current child process on the internal program stack, a new, unique program shell is created, at 44, to represent the child process. This shell is a program block that is dynamically attached to the calling parent process thorough a block hierarchy. Access to information and passing is through a block structure which wraps the shell." A person of ordinary skill in the art at the time of the invention would have been motivated to have a process call a child process because the child performs a function the parent needs to complete its execution. Wherein a procedure is some functionality implemented in a computer, it may be necessary, as shown by Holland, for that procedure to call another procedure to access additional functionality in order to complete its own functionality. Additionally, Examiner notes a procedure may call itself, i.e., recur, which also presents a parent-child relationship, and indeed, a common programming practice. Byers does not explicitly say that either of these conditions are present in the reference, i.e. a 102 type rejection, however, the need for a procedure to call another procedure or itself, creating a parent-child relationship and its associated stacks (as shown in Holland), is well known, and may be suggested by Byers in such figures as figure 2, where an agent communicates with another process; or figure 3, where a series of steps occur, wherein each step is implemented by some functionality, such functionality in a computer system being code; or figure 4, where, like figure 3, a series of steps occur, initiated by a shutdown; etc... Looking further at the abstract of Byers, a relationship between processes is disclosed, "application related processes", it being understood that a computer operates using program instructions, and that an application and related processes are in turn activated by preceding applications/processes/instructions, such as boostrap code, also disclosed in Byers. Byers further discloses that one process may terminate before another process that process related to another process (See figure 2, column 6, lines 22-36.), thereby affecting order of termination prior to shutdown. Whether this necessarily discloses a parent-child relationship is not clear, however, this relationship, and the manipulation of memory space by procedures in this relationship, is obviated by Holland, as shown above. Examiner further notes, that in at least this claim, no language is presented as to how one procedure is used by another procedure, no language is presented as to isolation from what, no language is presented as to isolation as it has to do with the "command".

4. Referring to claims 2-4, 14, 25, Byers in view of Holland discloses the external command is a terminate command, an interrupt command, and a resource-terminated command (From the abstract of Byers, "A processor shelf controller monitors processor shelf status. On shutdown or reset request either initiated by a central controller or the processor shelf, the processor shelf controller provides at least a first timer that allows the operating processor shelf sufficient time to shutdown operating system processes. For graceful shutdowns, a second timer is provided to allow the operating processor shelf to clear all application related processes prior to start of the first timer.").

5. Referring to claim 5, Byers in view of Holland discloses postponing execution of the external command for the predetermined period of time (From the abstract of Byers, "A processor shelf controller monitors processor shelf status. On shutdown or reset

request either initiated by a central controller or the processor shelf, the processor shelf controller provides at least a first timer that allows the operating processor shelf sufficient time to shutdown operating system processes. For graceful shutdowns, a second timer is provided to allow the operating processor shelf to clear all application related processes prior to start of the first timer.").

6. Referring to claim 6, although Byers in view of Holland does not specifically disclose the timer(s) may operate for a predefined period of time comprising a range of 4 to 6 seconds of CPU processing time, a length of time appropriate to a given task is well known in the art. Examiner takes official notice for 4 to 6 seconds. A person of ordinary skill in the art at the time of the invention would have been motivated to operate a timer for 4 to 6 seconds because, from line 21 of column 5 of Byers et al., "A number of software timers are used by the FSC 20 to track the progress of shelves 10 through shutdown and startup procedures. Timer periods are controlled by configuration data, and can be modified by the operations controller 24, should the operating characteristics of the system change. The following lists the key timers used in shelf supervision." and further from line 14 of column 6, "Graceful shutdown occurs in two stages. First the shelf 10 is allotted some time (typically several minutes) to clear out any service-related traffic. The FSC 20 marks this interval with a timer, the graceful shutdown timer 112. When the timer 112 expires, the FSC 20 issues a forced shutdown request 124 to the shelf 10. Assuming the timer 112 is properly matched to system characteristics, the shelf 10 will, by this time, have already begun to shut down its processes."

7. Referring to claim 7, 19, Byers in view of Holland discloses issuing a message to a system console (From line 52 of column 2 of Byers, "Shelves 10 periodically report their operational states and their alarm status to the FSC 10 via LSL 22.").

8. Referring to claim 8, 12, 17, Byers in view of Holland discloses terminating the first procedure and the second procedure (From the abstract of Byers, "A processor shelf controller monitors processor shelf status. On shutdown or reset request either initiated by a central controller or the processor shelf, the processor shelf controller provides at least a first timer that allows the operating processor shelf sufficient time to shutdown operating system processes. For graceful shutdowns, a second timer is provided to allow the operating processor shelf to clear all application related processes prior to start of the first timer.").

9. Referring to claim 9, Byers in view of Holland discloses executing the interrupt command (From the abstract of Byers, "A processor shelf controller monitors processor shelf status. On shutdown or reset request either initiated by a central controller or the processor shelf, the processor shelf controller provides at least a first timer that allows the operating processor shelf sufficient time to shutdown operating system processes. For graceful shutdowns, a second timer is provided to allow the operating processor shelf to clear all application related processes prior to start of the first timer.").

10. Referring to claim 10, Byers in view of Holland discloses receiving a command to increase a resource allocation amount by a predefined amount of time (From the abstract of Byers, "A processor shelf controller monitors processor shelf status. On shutdown or reset request either initiated by a central controller or the processor shelf,

the processor shelf controller provides at least a first timer that allows the operating processor shelf sufficient time to shutdown operating system processes. For graceful shutdowns, a second timer is provided to allow the operating processor shelf to clear all application related processes prior to start of the first timer.").

11. Referring to claim 11, Byers in view of Holland disclose postponing execution of the resource-terminated command for a specified period of time (From the abstract of Byers, "A processor shelf controller monitors processor shelf status. On shutdown or reset request either initiated by a central controller or the processor shelf, the processor shelf controller provides at least a first timer that allows the operating processor shelf sufficient time to shutdown operating system processes. For graceful shutdowns, a second timer is provided to allow the operating processor shelf to clear all application related processes prior to start of the first timer.")

12. Referring to claim 13, 29, Byers discloses a method for maintaining system integrity in a computer system, comprising: a first procedure (From the abstract (with emphasis), "A processor shelf controller monitors processor shelf status. On shutdown or reset request either initiated by a central controller or the processor shelf, the processor shelf controller provides at least a first timer that allows the operating processor shelf sufficient time to shutdown operating system **processes**."); and in response to a command associated with the first procedure, permitting the first procedure to continue processing for a predetermined period of time, before executing the command (From the abstract, "A processor shelf controller monitors processor shelf status. On shutdown or reset request either initiated by a central controller or the

processor shelf, the processor shelf controller provides at least a first timer that allows the operating processor shelf sufficient time to shutdown operating system processes. For graceful shutdowns, a second timer is provided to allow the operating processor shelf to clear all application related processes prior to start of the first timer.").

Although Byers does not specifically disclose associating a first procedure with a child stack, the first procedure having an associated second procedure, wherein the second procedure is a parent procedure and is associated with a parent stack, calling a child process by a parent is well known in the art. An example of this is shown by Holland, from line 9 of column 6 (with emphasis), "FIG. 3 shows the call flow for the compiler 12 of FIG. 1. First, at 40, the exemplary C external routine is called, for example, from the parent process 6 which provides a processor( ) shell port. Next, within the exemplary C context, at 42, the routine maintains a program stack of blocks to manage the transient execution of a plurality of child processes, such as 8,38 of FIG. 2B. The stack is employed because a parent process can call a child process, and that child process, in turn, can spawn other child processes using nested calls. As such, the program stack maintains **generated block information** so that program control can be returned in an orderly manner as child processes terminate, thereby returning to the respective parent processes. Then, after saving any current child process on the internal program stack, a new, unique program shell is created, at 44, to represent the child process. This shell is a program block that is dynamically attached to the calling parent process thorough a block hierarchy. Access to information and passing is through a block structure which wraps the shell." A person of ordinary skill in the art at the time of the invention would have been motivated to have a process call a child process because the child performs a function the parent needs to complete its execution. Wherein a procedure is some functionality implemented in a computer, it may be necessary, as shown by Holland, for that procedure to call another procedure to access additional functionality in order to complete its own functionality. Additionally, Examiner notes a procedure may call itself, i.e., recur, which also presents a parent-child relationship, and indeed, a common programming practice. Byers does not explicitly say that either of these conditions are present in the reference, i.e. a 102 type rejection, however, the need for a procedure to call another procedure or itself, creating a parent-child relationship and its associated stacks (as shown in Holland), is well known, and may be suggested by Byers in such figures as figure 2, where an agent communicates with another process; or figure 3, where a series of steps occur, wherein each step is implemented by some functionality, such functionality in a computer system being code; or figure 4, where, like figure 3, a series of steps occur, initiated by a shutdown; etc... Looking further at the abstract of Byers, a relationship between processes is disclosed, "application related processes", it being understood that a computer operates using program instructions, and that an application and related processes are in turn activated by preceding applications/processes/instructions, such as boostrap code, also disclosed in Byers. Byers further discloses that one process may terminate before another process that process related to another process (See figure 2, column 6, lines 22-36.), thereby affecting order of termination prior to shutdown. Whether this necessarily discloses a parent-child relationship is not clear, however, this relationship, and the manipulation of memory space by procedures in this relationship, is obviated by Holland, as shown above. Examiner further notes, that in at least this claim, no language is presented as to how one procedure is used by another procedure, no language is presented as to isolation from what, no language is presented as to isolation as it has to do with the "command".

13. Referring to claim 15, Byers in view of Holland discloses the interrupt command is issued because a time allotted to a resource has elapsed (From the abstract of Byers, "A processor shelf controller monitors processor shelf status. On shutdown or reset request either initiated by a central controller or the processor shelf, the processor shelf controller provides at least a first timer that allows the operating processor shelf sufficient time to shutdown operating system processes. For graceful shutdowns, a second timer is provided to allow the operating processor shelf to clear all application related processes prior to start of the first timer.").

14. Referring to claim 16, Byers in view of Holland discloses the time allotted to the resource is extended for a specified period of time (From line 49 of column 1 of Byers, "In accordance with another aspect of the present invention there is provided a method of controlling processor shelves comprising the steps of: auditing the status of the processor shelves; requesting shutdown of a particular processor shelf and starting a first timer; on expiry of the first timer, starting a second shutdown timer; and on expiry of the second timer sending a control signal to the processor shelf.").

15. Referring to claim 18, Byers in view of Holland discloses the interrupt command is executed (From the abstract of Byers, "A processor shelf controller monitors

processor shelf status. On shutdown or reset request either initiated by a central controller or the processor shelf, the processor shelf controller provides at least a first timer that allows the operating processor shelf sufficient time to shutdown operating system processes. For graceful shutdowns, a second timer is provided to allow the operating processor shelf to clear all application related processes prior to start of the first timer.").

16. Referring to claim 20, Byers in view of Holland discloses the child stack is comprised of at least one of a plurality of frames, wherein the at least one frame is associated with a procedure (From line 9 of column 6 of Holland (with emphasis), "FIG. 3 shows the call flow for the compiler 12 of FIG. 1. First, at 40, the exemplary C external routine is called, for example, from the parent process 6 which provides a processor( ) shell port. Next, within the exemplary C context, at 42, the routine maintains **a program stack of blocks** to manage the transient execution of a plurality of child processes, such as 8,38 of FIG. 2B. The stack is employed because a parent process can call a child process, and that child process, in turn, can spawn other child processes using nested calls. As such, the program stack maintains generated block information so that program control can be returned in an orderly manner as child processes terminate, thereby returning to the respective parent processes. Then, after saving any current child process on the internal program stack, a new, unique program shell is created, at 44, to represent the child process. This shell is a program block that is dynamically attached to the calling parent process thorough a block hierarchy. Access to information and passing is through a block structure which wraps the shell.").

17. Referring to claim 21, Byers in view of Holland discloses the at least one of a plurality of frames are processed in order from top to bottom (Holland specifically discloses a stack.).

18. Referring to claim 22, Byers in view of Holland discloses the at least one frame is marked isolated (From line 9 of column 6 of Holland (with emphasis), "FIG. 3 shows the call flow for the compiler 12 of FIG. 1. First, at 40, the exemplary C external routine is called, for example, from the parent process 6 which provides a processor( ) shell port. Next, within the exemplary C context, at 42, the routine maintains a program stack of blocks to manage the transient execution of a plurality of child processes, such as 8,38 of FIG. 2B. The stack is employed because a parent process can call a child process, and that child process, in turn, can spawn other child processes using nested calls. As such, the program stack maintains **generated block information** so that program control can be returned in an orderly manner as child processes terminate, thereby returning to the respective parent processes. Then, after saving any current child process on the internal program stack, a new, unique program shell is created, at 44, to represent the child process. This shell is a program block that is dynamically attached to the calling parent process thorough a block hierarchy. Access to information and passing is through a block structure which wraps the shell.").

19. Referring to claim 23, Byers discloses a method for maintaining system integrity in a computer system, comprising: first and second procedures (From the abstract (with emphasis), "A processor shelf controller monitors processor shelf status. On shutdown or reset request either initiated by a central controller or the processor shelf, the

processor shelf controller provides at least a first timer that allows the operating processor shelf sufficient time to shutdown operating system **processes**."); in response to receiving a terminate command associated with the second procedure, terminating the first procedure (From the abstract, "A processor shelf controller monitors processor shelf status. On shutdown or reset request either initiated by a central controller or the processor shelf, the processor shelf controller provides at least a first timer that allows the operating processor shelf sufficient time to shutdown operating system processes. For graceful shutdowns, a second timer is provided to allow the operating processor shelf to clear all application related processes prior to start of the first timer." Wherein the second process is related to the first process at least insomuch as they are "application related proceses" (see abstract), and a shelf termination command, as disclosed in Byers, terminates all procedures.).

Although Byers does not specifically disclose associating a first procedure with a child stack, the first procedure having an associated second procedure, wherein the second procedure is a parent procedure and is associated with a parent stack, calling a child process by a parent is well known in the art. An example of this is shown by Holland, from line 9 of column 6 (with emphasis), "FIG. 3 shows the call flow for the compiler 12 of FIG. 1. First, at 40, the exemplary C external routine is called, for example, from the parent process 6 which provides a processor( ) shell port. Next, within the exemplary C context, at 42, the routine maintains a program stack of blocks to manage the transient execution of a plurality of child processes, such as 8,38 of FIG. 2B. The stack is employed because a parent process can call a child process, and that child process, in turn, can spawn other child processes using nested calls. As such, the program stack maintains **generated block information** so that program control can be returned in an orderly manner as child processes terminate, thereby returning to the respective parent processes. Then, after saving any current child process on the internal program stack, a new, unique program shell is created, at 44, to represent the child process. This shell is a program block that is dynamically attached to the calling parent process thorough a block hierarchy. Access to information and passing is through a block structure which wraps the shell." A person of ordinary skill in the art at the time of the invention would have been motivated to have a process call a child process because the child performs a function the parent needs to complete its execution. Wherein a procedure is some functionality implemented in a computer, it may be necessary, as shown by Holland, for that procedure to call another procedure to access additional functionality in order to complete its own functionality. Additionally, Examiner notes a procedure may call itself, i.e., recur, which also presents a parent-child relationship, and indeed, a common programming practice. Byers does not explicitly say that either of these conditions are present in the reference, i.e. a 102 type rejection, however, the need for a procedure to call another procedure or itself, creating a parent-child relationship and its associated stacks (as shown in Holland), is well known, and may be suggested by Byers in such figures as figure 2, where an agent communicates with another process; or figure 3, where a series of steps occur, wherein each step is implemented by some functionality, such functionality in a computer system being code; or figure 4, where, like figure 3, a series of steps occur, initiated by a shutdown; etc... Looking further at the abstract of Byers, a relationship between processes is disclosed, "application related processes", it being understood that a computer operates using program instructions, and that an application and related processes are in turn activated by preceding applications/processes/instructions, such as boostrap code, also disclosed in Byers. Byers further discloses that one process may terminate before another process that process related to another process (See figure 2, column 6, lines 22-36.), thereby affecting order of termination prior to shutdown. Whether this necessarily discloses a parent-child relationship is not clear, however, this relationship, and the manipulation of memory space by procedures in this relationship, is obviated by Holland, as shown above. Examiner further notes, that in at least this claim, no language is presented as to how one procedure is used by another procedure, no language is presented as to isolation from what, no language is presented as to isolation as it has to do with the "command".

20. Referring to claim 24, Byers discloses a system for maintaining system integrity comprising: and a central processing unit that executes computer-readable instructions for maintaining system integrity in a multiple user environment (From line 35 of column 2, "LAN".), the computer-readable instructions including instructions for: a first procedure (From the abstract (with emphasis), "A processor shelf controller monitors processor shelf status. On shutdown or reset request either initiated by a central controller or the processor shelf, the processor shelf controller provides at least a first timer that allows the operating processor shelf sufficient time to shutdown operating system **processes**."), and in response to receiving a command associated with the first

procedure, before executing the command, permitting the first procedure to continue processing for a predetermined period of time (From the abstract, "A processor shelf controller monitors processor shelf status. On shutdown or reset request either initiated by a central controller or the processor shelf, the processor shelf controller provides at least a first timer that allows the operating processor shelf sufficient time to shutdown operating system processes. For graceful shutdowns, a second timer is provided to allow the operating processor shelf to clear all application related processes prior to start of the first timer.").

Although Byers does not specifically disclose a memory for storing and manipulating stacks; and associating a first procedure with a child stack, the first procedure having an associated second procedure, wherein the second procedure is a parent procedure and is associated with a parent stack, calling a child process by a parent is well known in the art. An example of this is shown by Holland, from line 9 of column 6 (with emphasis), "FIG. 3 shows the call flow for the compiler 12 of FIG. 1. First, at 40, the exemplary C external routine is called, for example, from the parent process 6 which provides a processor( ) shell port. Next, within the exemplary C context, at 42, the routine maintains a program stack of blocks to manage the transient execution of a plurality of child processes, such as 8,38 of FIG. 2B. The stack is employed because a parent process can call a child process, and that child process, in turn, can spawn other child processes using nested calls. As such, the program stack maintains **generated block information** so that program control can be returned in an orderly manner as child processes terminate, thereby returning to the respective parent processes. Then, after saving any current child process on the internal program stack, a new, unique program shell is created, at 44, to represent the child process. This shell is a program block that is dynamically attached to the calling parent process thorough a block hierarchy. Access to information and passing is through a block structure which wraps the shell." A person of ordinary skill in the art at the time of the invention would have been motivated to have a process call a child process because the child performs a function the parent needs to complete its execution. Wherein a procedure is some functionality implemented in a computer, it may be necessary, as shown by Holland, for that procedure to call another procedure to access additional functionality in order to complete its own functionality. Additionally, Examiner notes a procedure may call itself, i.e., recur, which also presents a parent-child relationship, and indeed, a common programming practice. Byers does not explicitly say that either of these conditions are present in the reference, i.e. a 102 type rejection, however, the need for a procedure to call another procedure or itself, creating a parent-child relationship and its associated stacks (as shown in Holland), is well known, and may be suggested by Byers in such figures as figure 2, where an agent communicates with another process; or figure 3, where a series of steps occur, wherein each step is implemented by some functionality, such functionality in a computer system being code; or figure 4, where, like figure 3, a series of steps occur, initiated by a shutdown; etc... Looking further at the abstract of Byers, a relationship between processes is disclosed, "application related processes", it being understood that a computer operates using program instructions, and that an application and related processes are in turn activated by preceding applications/processes/instructions, such as boostrap code, also disclosed in Byers. Byers further discloses that one process may terminate before another process that process related to another process (See figure 2, column 6, lines 22-36.), thereby affecting order of termination prior to shutdown. Whether this necessarily discloses a parent-child relationship is not clear, however, this relationship, and the manipulation of memory space by procedures in this relationship, is obviated by Holland, as shown above. Examiner further notes, that in at least this claim, no language is presented as to how one procedure is used by another procedure, no language is presented as to isolation from what, no language is presented as to isolation as it has to do with the "command".

21. Referring to claim 26, Byers in view of Holland discloses the computer-readable instructions comprise further computer-readable instructions to terminate the first procedure and the second procedure if the first procedure does not complete execution within the predetermined period of time (From the abstract of Byers, "A processor shelf controller monitors processor shelf status. On shutdown or reset request either initiated by a central controller or the processor shelf, the processor shelf controller provides at least a first timer that allows the operating processor shelf sufficient time to shutdown operating system processes. For graceful shutdowns, a second timer is provided to allow the operating processor shelf to clear all application related processes prior to start of the first timer.").

22. Referring to claim 27, Byers in view of Holland discloses the command associated with the first procedure is the command to interrupt the first procedure (From

the abstract of Byers, "A processor shelf controller monitors processor shelf status. On shutdown or reset request either initiated by a central controller or the processor shelf, the processor shelf controller provides at least a first timer that allows the operating processor shelf sufficient time to shutdown operating system processes. For graceful shutdowns, a second timer is provided to allow the operating processor shelf to clear all application related processes prior to start of the first timer.").

23. Referring to claim 28, Byers in view of Holland discloses computer-readable instructions for interrupting the first procedure if the first procedure does not complete execution within the predetermined period of time (From the abstract of Byers, "A processor shelf controller monitors processor shelf status. On shutdown or reset request either initiated by a central controller or the processor shelf, the processor shelf controller provides at least a first timer that allows the operating processor shelf sufficient time to shutdown operating system processes. For graceful shutdowns, a second timer is provided to allow the operating processor shelf to clear all application related processes prior to start of the first timer.").

Conclusion

24. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

US 5815702 to Kannan et al.

US 5835763 to Klein

US 5872981 to Waddington et al.

US 6269478 to Lautenbach-Lampe et al.

US 6553512 to Gibson

JP401213727A to Otsu

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Gabriel L. Chu whose telephone number is (571) 272-3656. The examiner can normally be reached on weekdays between 8:30 AM and 5:00 PM.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Robert W. Beausoliel, Jr. can be reached on (571) 272-3645. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

[illegible]

ROBERT BEAUSOLIEL
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100

gc